ALPHAREFINE: OPACITY-ROUTED REFINEMENT FOR FEED-FORWARD DRIVING SCENE RECONSTRUCTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Feed-forward driving scene reconstruction has emerged as a scalable alternative to per-scene optimization, but still struggles to produce globally consistent appearance and fine structural details. Recent efforts close this gap through diffusion-based post-hoc refinement, which applies a uniform and computationally heavy generative enhancement. However, this uniform treatment overlooks the actual error structure of feed-forward renderings. Through empirical analysis on multiple feed-forward models, we observe that rendering errors are not spatially homogeneous but strongly correlated with accumulated opacity, where low-opacity regions are dominated by low-frequency appearance shifts and high-opacity regions suffer primarily from high-frequency structural loss. Building on this observation, we identify a complementary correspondence between these two error regimes and two in-context correction signals encoded in the source view. Specifically, the source image serves as appearance context for low-frequency correction, while the source-view residual provides error context for high-frequency recovery. Based on this insight, we introduce **AlphaRefine**, a lightweight post-hoc refinement framework that routes the two corrections by accumulated opacity, applicable to any frozen feed-forward driving reconstruction model. Under the same evaluation protocols, our method achieves new state-of-the-art among feed-forward methods on both Waymo and nuScenes, boosting STORM to 28.32 dB ($+2.51$ dB) and DrivingForward to 29.83 dB ($+2.46$ dB), while running over $9\times$ faster than single-step diffusion refiners at only 6 ms per image. Code will be released.

1 INTRODUCTION

Reconstructing dynamic street environments from on-board multi-camera observations is a fundamental challenge in autonomous driving. A high-fidelity, renderable, and simulatable 4D representation is essential for unlocking advanced closed-loop simulation and downstream planning. Earlier reconstruction pipelines typically rely on scene-specific optimization, which can achieve strong visual quality but remains too slow for practical use at autonomous-driving scale (Tancik et al., 2022; Yang et al., 2024; Yan et al., 2024; Zhou et al., 2024). This bottleneck has motivated a growing shift toward feed-forward 4D reconstruction, where a single model directly predicts the scene representation and novel-view renderings in one forward pass (Tian et al., 2025; Lu et al., 2025; Yang et al., 2025; Chen et al., 2025a; Shi et al., 2026).

Despite their efficiency, feed-forward models still struggle to match the rendering fidelity of optimized methods, owing to the challenge of simultaneously preserving globally consistent appearance and recovering fine structures without scene-specific optimization. To bridge this quality gap, recent efforts often resort to diffusion models for post-hoc refinement (Wu et al., 2025; Chen et al., 2025a), which apply a *uniform* and computationally heavy generative enhancement across the entire image. However, through a comprehensive empirical analysis across multiple feed-forward driving reconstruction models, we demonstrate that these errors are fundamentally not spatially homogeneous. Instead, they are strongly correlated with accumulated opacity. As supported by Fig. 1, regions with low accumulated opacity are predominantly prone to low-frequency appearance degradation, such as color drift and luminance inconsistency. Conversely, regions with high accumulated opacity suffer primarily from the loss of high-frequency structures and fine details.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

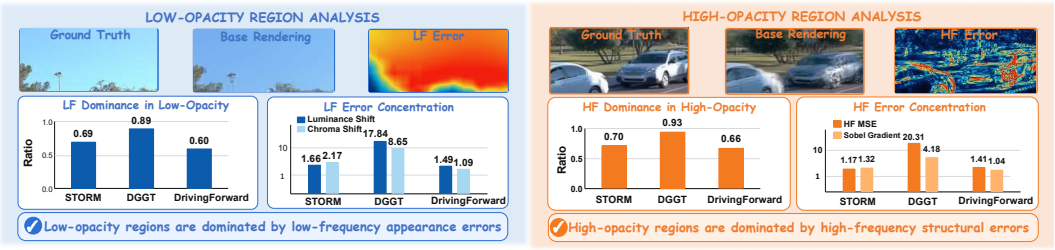

Figure 1: Spatially heterogeneous rendering errors in feed-forward driving reconstruction. Low-opacity regions are dominated by low-frequency appearance errors, while high-opacity regions are dominated by high-frequency structural errors.

Given this spatially varying degradation, applying a *uniform* generative enhancement across the entire image is inherently suboptimal. Resolving these distinct error types demands an adaptive, decoupled strategy. To the best of our knowledge, we are the first to identify a novel correspondence between the target (new) view's opacity-dependent error regimes and two distinct in-context correction signals naturally encoded in the source (reference) view. Specifically, the raw source image provides appearance context for correcting low-frequency color deviations in low-opacity regions. By contrast, the source-view residual, defined as the difference between the source-view ground truth and its initial rendering (Rong et al., 2022), provides error context that explicitly captures what the base reconstruction model fails to preserve. This makes the residual particularly suitable for compensating the loss of high-frequency structures and fine details in high-opacity regions.

Building upon this complementary relationship, we introduce **AlphaRefine**, a general and lightweight post-hoc refinement framework that seamlessly attaches to diverse feed-forward driving reconstruction models without altering their internal architectures. Given the initial outputs of a frozen base model, our framework employs a dual-branch design that uses the accumulated opacity as a natural gating mechanism to softly modulate their respective contributions. Specifically, the first branch, *CVAC*, leverages the source image as a global appearance context to rectify low-frequency color and luminance deviations primarily in low-opacity areas. Concurrently, the second branch, *CVRC*, constructs the source-view residual as error context and geometrically transports it to the target view to recover fine structures in high-opacity regions. Because such cross-view transport is inherently vulnerable to occlusions and dynamic objects, we introduce a small dynamic correction module, *DCN*, as a crucial safeguard. This module adaptively adjusts sampling locations and suppresses unreliable residual features to guarantee robust structural recovery. By unifying these targeted corrections, our framework resolves heterogeneous rendering errors efficiently in feed-forward driving reconstruction, completely bypassing the severe computational overhead and hallucination risks associated with heavy diffusion-based refinement.

We extensively evaluate our framework on the Waymo (Sun et al., 2020) and nuScenes (Caesar et al., 2020) datasets by plugging it into leading feed-forward driving reconstruction models, including STORM (Yang et al., 2025), DGGT (Chen et al., 2025a), and DrivingForward (Tian et al., 2025). The results demonstrate consistent improvements in novel-view synthesis. Most notably, our method establishes new state-of-the-art performance among feed-forward methods, with STORM reaching 28.32 dB PSNR on Waymo (+2.51 dB) and DrivingForward reaching 29.83 dB on nuScenes (+2.46 dB). All evaluated models also achieve substantial gains in PSNR, SSIM, and LPIPS on both datasets. Beyond absolute rendering quality, our framework exhibits overwhelming advantages in efficiency when compared to recent diffusion-based refiners. While explicitly and surprisingly outperforming them in visual fidelity, our pipeline requires substantially less fine-tuning and runs over $9\times$ faster than even single-step diffusion refinement. Taking a mere 6 ms per image, our lightweight refinement introduces practically zero computational overhead to the base models, making it a highly scalable and practical solution for real-time autonomous driving applications.

Our contributions are: ♠ **Novel Insight.** We first identify a fundamental correspondence between the target view's opacity-dependent error regimes and two complementary in-context correction signals encoded in the source views. ♠ **General Framework.** We propose **AlphaRefine**, a lightweight post-hoc refinement pipeline that exploits source-view appearance and error context to resolve heterogeneous rendering errors. By employing an opacity-routed dual-branch architecture, it adap-

tively applies targeted corrections without altering the internal structures of the base models. ♠ **New State-of-the-Art Performance and Efficiency.** Extensive experiments demonstrate that our method consistently improves multiple base models and sets a new state-of-the-art performance. Furthermore, it outperforms diffusion refiners in visual fidelity while requiring substantially fewer fine-tuning steps and running over $9\times$ faster than even single-step diffusion refinement.

2 Related Work

2.1 Driving Scene Reconstruction

Per-scene optimization. Reconstructing autonomous driving scenes is challenging due to large unbounded extents, sparse forward-facing views, and dynamic traffic participants entangled with the static background. Early efforts extend Neural Radiance Fields (Mildenhall et al., 2020) to street-scale dynamic environments (Tancik et al., 2022; Rematas et al., 2022; Xie et al., 2023; Ost et al., 2021; Yang et al., 2023; 2024). Following the emergence of 3D Gaussian Splatting (Kerbl et al., 2023), per-scene optimization methods have been proposed: DrivingGaussian (Zhou et al., 2024) leverages composite Gaussian graphs and LiDAR priors, Street Gaussians (Yan et al., 2024) introduces 4D spherical harmonics with tracked pose optimization, PVG (Chen et al., 2023) models per-Gaussian dynamic properties through periodic vibration, and OmniRe (Chen et al., 2025b) extends to non-rigid actors via deformable Gaussians. Despite achieving strong reconstruction quality, these methods rely on per-scene optimization that takes minutes to hours per sequence, which is computationally prohibitive for large-scale autonomous driving applications.

Feed-forward Reconstruction. Feed-forward methods replace per-scene optimization with a single network pass that directly predicts a 3D or 4D scene representation. Recent works adapt this paradigm to autonomous driving scenes (Lu et al., 2025; Miao et al., 2025; Wang et al., 2025; Tian et al., 2025; Yang et al., 2025; Chen et al., 2025a; Shi et al., 2026). For example, DrivingForward (Tian et al., 2025) jointly trains pose, depth, and Gaussian networks to reconstruct surround-view scenes from sparse inputs in a self-supervised manner. STORM (Yang et al., 2025) predicts per-frame 3D Gaussians together with self-supervised scene flow via motion tokens, enabling real-time reconstruction of dynamic scenes. DGGT (Chen et al., 2025a) proposes a pose-free framework that jointly infers Gaussian maps, camera parameters, and dynamic motion in a single pass. While these methods reduce reconstruction time from hours to seconds, they still struggle to preserve consistent appearance and fine structural details. Rather than designing yet another feed-forward architecture, we introduce a lightweight post-hoc refinement framework that can be applied to any existing feed-forward driving model to improve its rendering quality.

2.2 Rendering Refinement

A complementary line of work resorts to diffusion priors to enhance the rendered images of a reconstructed scene (Wu et al., 2025; de Lutio et al., 2026; Yin et al., 2025). Difix3D+ (Wu et al., 2025) trains a single-step image diffusion model that removes artifacts from rendered novel views. ArtiFixer (de Lutio et al., 2026) employs opacity-guided video diffusion and distills it into a causal autoregressive model to generate long, consistent novel-view sequences. GSFixer (Yin et al., 2025) leverages a reference-guided video diffusion model that incorporates semantic and geometric features from source views to restore artifacts in novel-view renderings. Furthermore, DGGT (Chen et al., 2025a) adopts Difix3D+ as a refinement step in its feed-forward pipeline. While achieving impressive results, these diffusion-based refiners apply heavy generative enhancement uniformly across the entire image, incur substantial inference cost, and may introduce hallucinated details inconsistent with the original rendering. In contrast, we propose a lightweight refinement framework that addresses heterogeneous errors with two complementary source-view signals, achieving higher visual fidelity at only 6 ms per image.

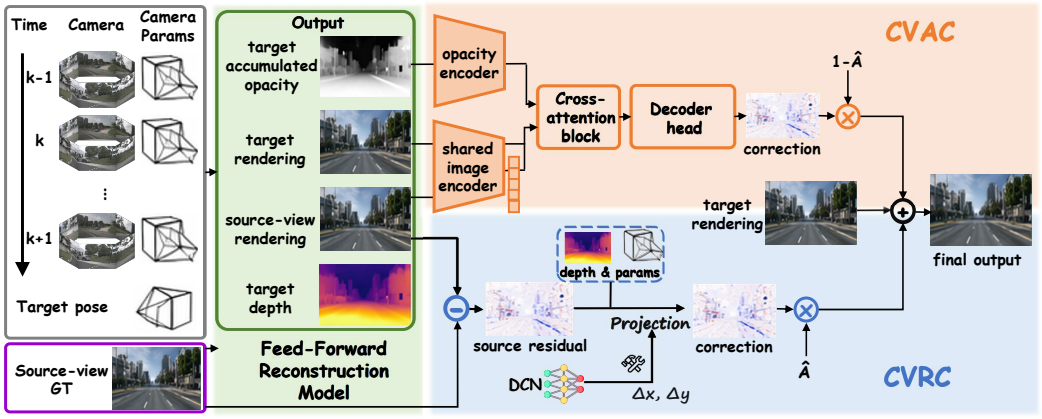

Figure 2: Overview of our opacity-routed refinement framework. Given the initial outputs of a frozen feed-forward model, CVAC corrects low-frequency appearance errors gated by $(1 - \hat{A}_{\text{tgt}})$, and CVRC transports the source-view residual gated by $\hat{A}_{\text{tgt}}$ to recover high-frequency details, with DCN adjusting the sampling location in dynamic regions.

3 METHOD

3.1 OVERVIEW

Problem formulation. We consider feed-forward novel-view synthesis for autonomous driving scenes. The input consists of a spatio-temporal image sequence $\mathcal{I} = \{\mathbf{I}^{t,k}\}_{t=1,\dots,T;\, k=1,\dots,K}$ and the corresponding camera parameters $\mathcal{P} = \{\mathbf{P}^{t,k}\}$, where t indexes the T timesteps and k indexes the K on-board cameras. Given target camera parameters $\mathbf{P}_{\text{tgt}}$, a feed-forward reconstruction model $f_{\boldsymbol{\Theta}}$ predicts a target-view rendering $\hat{\mathbf{I}}_{\text{tgt}} = f_{\boldsymbol{\Theta}}(\mathcal{I}, \mathcal{P}, \mathbf{P}_{\text{tgt}}) \in \mathbb{R}^{H \times W \times 3}$, where H and W denote image height and width. Our method further refines $\hat{\mathbf{I}}_{\text{tgt}}$ to improve the rendering quality.

Pipeline. AlphaRefine starts from a frozen feed-forward reconstruction model and refines its target-view rendering $\hat{\mathbf{I}}_{\text{tgt}}$ in two successive stages. To support both stages, we further query the model for two auxiliary outputs at the target view, the predicted depth $\hat{\mathbf{D}}_{\text{tgt}} \in \mathbb{R}^{H \times W}$ and the accumulated opacity $\hat{\mathbf{A}}_{\text{tgt}} \in [0, 1]^{H \times W}$, defined by

$$\hat{\mathbf{A}}_{\text{tgt}} = \sum_{j=1}^{N} \alpha_j \prod_{l=1}^{j-1} (1 - \alpha_l)\,, \tag{1}$$

where α_j is the opacity of the j-th Gaussian at the pixel and N is the number of Gaussians.

The first branch, CVAC, is a small neural network $F_{\boldsymbol{\theta}}$ parameterized by $\boldsymbol{\theta}$ that takes the target rendering $\hat{\mathbf{I}}_{\text{tgt}}$, a source-view image $\mathbf{I}_{\text{src}}$, and the target accumulated opacity $\hat{\mathbf{A}}_{\text{tgt}}$ as inputs, and predicts a correction whose strength is gated by $(1 - \hat{\mathbf{A}}_{\text{tgt}})$ and a learnable global scalar s,

$$\hat{\mathbf{I}}^{\text{ref}}_{\text{tgt}} = \hat{\mathbf{I}}_{\text{tgt}} + s \cdot (1 - \hat{\mathbf{A}}_{\text{tgt}}) \odot F_{\boldsymbol{\theta}}\Big(\hat{\mathbf{I}}_{\text{tgt}},\, \mathbf{I}_{\text{src}},\, \hat{\mathbf{A}}_{\text{tgt}}\Big)\,. \tag{2}$$

The second branch, CVRC, transports the source-view error context to the target view to recover high-frequency details. To this end, we render the model at the source view to obtain $\hat{\mathbf{I}}_{\text{src}}$ and construct the source-view residual $\mathbf{R}_{\text{src}} = \mathbf{I}_{\text{src}} - \hat{\mathbf{I}}_{\text{src}}$. We then reproject the target view onto the source view through $\hat{\mathbf{D}}_{\text{tgt}}$ and the camera parameters, yielding a base sampling coordinate $\mathbf{p}^{\text{base}}_{\text{src}}$. To handle dynamic regions where the static-correspondence assumption breaks down, we introduce DCN, a lightweight network that predicts an offset $\delta\mathbf{p}$ to correct the sampling location. The transported residual is bilinearly sampled from $\mathbf{R}_{\text{src}}$ and added to $\hat{\mathbf{I}}^{\text{ref}}_{\text{tgt}}$ via opacity-based modulation,

$$\hat{\mathbf{I}}^{\text{final}}_{\text{tgt}} = \hat{\mathbf{I}}^{\text{ref}}_{\text{tgt}} + \hat{\mathbf{A}}_{\text{tgt}} \odot \lambda \cdot \text{Sample}\big(\mathbf{R}_{\text{src}},\, \mathbf{p}^{\text{base}}_{\text{src}} + \delta\mathbf{p}\big)\,, \tag{3}$$

where λ is a global weight. Fig. 2 illustrates the schematic overview.

3.2 CVAC: CROSS-VIEW APPEARANCE CORRECTION

Module design. In the target view, low-$\hat{\mathbf{A}}_{\mathrm{tgt}}$ regions such as sky and distant background suffer primarily from low-frequency errors such as color drift and luminance inconsistency, as analyzed in Fig. 1. To correct these errors, we design CVAC as a lightweight network that uses $\mathbf{I}_{\mathrm{src}}$ as appearance context. Specifically, CVAC takes the target rendering $\hat{\mathbf{I}}_{\mathrm{tgt}}$, the source-view image $\mathbf{I}_{\mathrm{src}}$, and the target opacity $\hat{\mathbf{A}}_{\mathrm{tgt}}$ as inputs, and predicts a correction $\Delta\mathbf{I}_{\mathrm{tgt}} = F_{\boldsymbol{\theta}}(\hat{\mathbf{I}}_{\mathrm{tgt}}, \mathbf{I}_{\mathrm{src}}, \hat{\mathbf{A}}_{\mathrm{tgt}})$. We further gate the correction strength by $(1 - \hat{\mathbf{A}}_{\mathrm{tgt}})$ together with a learnable global scalar s, so that CVAC is primarily active in low-$\hat{\mathbf{A}}_{\mathrm{tgt}}$ regions, yielding the refined rendering

$$\hat{\mathbf{I}}^{\mathrm{ref}}_{\mathrm{tgt}} = \hat{\mathbf{I}}_{\mathrm{tgt}} + s \cdot \big(1 - \hat{\mathbf{A}}_{\mathrm{tgt}}\big) \odot \Delta\mathbf{I}_{\mathrm{tgt}}. \tag{4}$$

Network architecture. CVAC consists of four components, a shared image encoder, an opacity encoder, a cross-attention block, and a decoder that predicts the target-view correction $\Delta\mathbf{I}_{\mathrm{tgt}}$. The target rendering $\hat{\mathbf{I}}_{\mathrm{tgt}}$ and source-view image $\mathbf{I}_{\mathrm{src}}$ are embedded into $\mathbf{F}_{\mathrm{tgt}}$ and $\mathbf{F}_{\mathrm{src}}$ by a shared image encoder, while $\hat{\mathbf{A}}_{\mathrm{tgt}}$ is encoded into $\mathbf{F}_A$ with a separate opacity encoder. The aggregated source feature is then obtained by temporal averaging over the same-camera source-view frames, $\bar{\mathbf{F}}_{\mathrm{src}} = \frac{1}{M}\sum_{n=1}^{M} \mathbf{F}_{\mathrm{src},n}$, where M is the number of source frames. The target feature then queries appearance context from the aggregated source feature through a cross-attention block, yielding $\mathbf{F}_{\mathrm{corr}} = \mathrm{CrossAttn}(\mathbf{F}_{\mathrm{tgt}}, \bar{\mathbf{F}}_{\mathrm{src}}, \bar{\mathbf{F}}_{\mathrm{src}})$ where $\mathbf{F}_{\mathrm{tgt}}$ acts as the query and $\bar{\mathbf{F}}_{\mathrm{src}}$ serves as both key and value. Finally, the target feature, the cross-attended source feature, and the opacity feature are concatenated along the channel and passed to a lightweight decoder H to predict the correction, $\Delta\mathbf{I}_{\mathrm{tgt}} = H([\mathbf{F}_{\mathrm{tgt}}, \mathbf{F}_{\mathrm{corr}}, \mathbf{F}_A])$. In total, CVAC contains only 1M parameters, making it lightweight and efficient at inference.

3.3 CVRC: CROSS-VIEW RESIDUAL CORRECTION

Module design. To complement CVAC's low-frequency corrections, CVRC leverages the error context observed at the source view to compensate for high-frequency loss in the target view. Given the source residual $\mathbf{R}_{\mathrm{src}}$, we obtain its transported counterpart $\tilde{\mathbf{R}}_{\mathrm{src}\to\mathrm{tgt}}$ via geometric reprojection, explicitly preserving high-frequency structural details. Since the structural errors targeted by CVRC are concentrated in high-$\hat{\mathbf{A}}_{\mathrm{tgt}}$ regimes, we further gate the transport by the accumulated opacity. Ultimately, the two branches employ these complementary gates, $(1 - \hat{\mathbf{A}}_{\mathrm{tgt}})$ for CVAC and $\hat{\mathbf{A}}_{\mathrm{tgt}}$ for CVRC, to jointly resolve the heterogeneous errors in the target rendering.

Residual transport. We now specify how $\tilde{\mathbf{R}}_{\mathrm{src}\to\mathrm{tgt}}$ is constructed from $\mathbf{R}_{\mathrm{src}}$. For a target pixel $\mathbf{p}_t$, we first lift it to a 3D point in the world frame using the predicted depth and the target camera parameters, $\mathbf{X}_w = \mathbf{T}_{\mathrm{tgt}}\,\hat{\mathbf{D}}_{\mathrm{tgt}}(\mathbf{p}_t)\,\mathbf{K}_{\mathrm{tgt}}^{-1}\,\tilde{\mathbf{p}}_t$, where $\tilde{\mathbf{p}}_t$ denotes $\mathbf{p}_t$ in homogeneous coordinates, and $\mathbf{K}_{\mathrm{tgt}}$ and $\mathbf{T}_{\mathrm{tgt}}$ are the target intrinsics and extrinsics. We then project $\mathbf{X}_w$ onto the source image plane through the source camera parameters to obtain the sampling coordinate $\mathbf{p}^{\mathrm{base}}_{\mathrm{src}} = \Pi_{\mathrm{src}}(\mathbf{X}_w)$. The transported residual at $\mathbf{p}_t$ is obtained by bilinearly sampling $\mathbf{R}_{\mathrm{src}}$ at this coordinate, yielding

$$\tilde{\mathbf{R}}_{\mathrm{src}\to\mathrm{tgt}}(\mathbf{p}_t) = \mathrm{Sample}\big(\mathbf{R}_{\mathrm{src}},\, \mathbf{p}^{\mathrm{base}}_{\mathrm{src}}\big)\,. \tag{5}$$

The transported residual is then added back to the refined target rendering with opacity-based modulation, yielding $\hat{\mathbf{I}}^{\mathrm{cvrc}}_{\mathrm{tgt}} = \hat{\mathbf{I}}^{\mathrm{ref}}_{\mathrm{tgt}} + \hat{\mathbf{A}}_{\mathrm{tgt}} \odot \lambda \cdot \tilde{\mathbf{R}}_{\mathrm{src}\to\mathrm{tgt}}$. The transport defined above assumes static correspondence between the two views, which becomes inaccurate in dynamic regions. We next introduce a dynamic correction to account for this mismatch.

Dynamic correction. To address mismatches in dynamic regions, we introduce the Dynamic Correction Network (DCN), a lightweight module $G_{\boldsymbol{\psi}}$ that predicts a sampling offset for the source view. This network takes a 9-channel tensor as input, formed by concatenating the source image, the target view reprojected onto the source view, and their absolute difference. It predicts a 2D offset $\delta\mathbf{p} = (\Delta x, \Delta y) = G_{\boldsymbol{\psi}}([\mathbf{I}_{\mathrm{src}}, \hat{\mathbf{I}}_{\mathrm{tgt}\to\mathrm{src}}, |\mathbf{I}_{\mathrm{src}} - \hat{\mathbf{I}}_{\mathrm{tgt}\to\mathrm{src}}|])$. We then compute the corrected sampling coordinate as $\mathbf{p}_{\mathrm{src}} = \mathbf{p}^{\mathrm{base}}_{\mathrm{src}} + \delta\mathbf{p}$. Using this updated location, the dynamically corrected transported residual is calculated via bilinear sampling

$$\tilde{\mathbf{R}}^{\mathrm{dyn}}_{\mathrm{src}\to\mathrm{tgt}}(\mathbf{p}_t) = \mathrm{Sample}\big(\mathbf{R}_{\mathrm{src}},\, \mathbf{p}^{\mathrm{base}}_{\mathrm{src}} + \delta\mathbf{p}\big)\,. \tag{6}$$

Table 1: Main quantitative results on Waymo and nuScenes, where Δ denotes the improvement of our method over the baseline. Our refinement consistently improves all three feed-forward models on both datasets, setting new state-of-the-art on each benchmark.

	Waymo			nuScenes		
Method	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
STORM	25.81	0.804	0.099	22.10	0.656	0.175
STORM + Ours	28.32	0.850	0.074	25.37	0.740	0.134
Δ	*+2.51*	*+0.046*	*-0.025*	*+3.27*	*+0.084*	*-0.041*
DGGT	24.64	0.799	0.104	21.58	0.795	0.134
DGGT + Ours	27.10	0.825	0.072	24.32	0.818	0.095
Δ	*+2.46*	*+0.026*	*-0.032*	*+2.74*	*+0.023*	*-0.039*
DrivingForward	24.85	0.764	0.116	27.37	0.820	0.084
DrivingForward + Ours	28.07	0.809	0.080	29.83	0.852	0.062
Δ	*+3.22*	*+0.045*	*-0.036*	*+2.46*	*+0.032*	*-0.022*

The final target rendering is then updated as

$$\hat{\mathbf{I}}_{\mathrm{tgt}}^{\mathrm{final}} = \hat{\mathbf{I}}_{\mathrm{tgt}}^{\mathrm{ref}} + \hat{\mathbf{A}}_{\mathrm{tgt}} \odot \lambda \cdot \tilde{\mathbf{R}}_{\mathrm{src}\to\mathrm{tgt}}^{\mathrm{dyn}}. \tag{7}$$

In this manner, CVRC extends geometric residual transport to a dynamic-aware correction mechanism that remains compatible with the overall refinement pipeline.

3.4 Training

Joint training. We train the entire refinement stack jointly, with CVAC and DCN trainable, while keeping the feed-forward reconstruction model frozen. During training, the model first provides the target-view rendering $\hat{\mathbf{I}}_{\mathrm{tgt}}$, the source-view rendering $\hat{\mathbf{I}}_{\mathrm{src}}$, and the auxiliary target-view predictions, the depth $\hat{\mathbf{D}}_{\mathrm{tgt}}$ and accumulated opacity $\hat{\mathbf{A}}_{\mathrm{tgt}}$. These outputs are then passed through the full refinement pipeline. CVAC first produces the refined rendering $\hat{\mathbf{I}}_{\mathrm{tgt}}^{\mathrm{ref}}$, after which CVRC and DCN perform dynamic-aware residual transport to obtain the final output $\hat{\mathbf{I}}_{\mathrm{tgt}}^{\mathrm{final}}$. In this manner, the entire framework is optimized end-to-end.

Training losses. Our training objective consists of three terms, an intermediate image reconstruction loss on the CVAC output, a final reconstruction loss on the fully refined rendering, and an offset supervision loss for DCN. The two image reconstruction losses are applied to $\hat{\mathbf{I}}_{\mathrm{tgt}}^{\mathrm{ref}}$ and $\hat{\mathbf{I}}_{\mathrm{tgt}}^{\mathrm{final}}$, denoted $\mathcal{L}_{\mathrm{ref}}$ and $\mathcal{L}_{\mathrm{final}}$, respectively, where each is a weighted combination of pixel-wise L_1 loss, SSIM loss, and perceptual loss. For the offset supervision $\mathcal{L}_{\mathrm{dcn}}$, we derive the reference targets by projecting the per-Gaussian motion predictions from STORM (Yang et al., 2025) onto the image plane. As a result, DCN is jointly supervised by the image reconstruction loss and the explicit offset signal. The full training objective is written as

$$\mathcal{L} = \lambda_{\mathrm{ref}}\mathcal{L}_{\mathrm{ref}} + \lambda_{\mathrm{final}}\mathcal{L}_{\mathrm{final}} + \lambda_{\mathrm{dcn}}\mathcal{L}_{\mathrm{dcn}}, \tag{8}$$

where $\lambda_{\mathrm{ref}}, \lambda_{\mathrm{final}}$ and λ_{dcn} are scalar weights.

Transfer to other reconstruction models. We first train the full refinement framework on STORM for 100K steps using the joint objective in Eq. 8. This initial stage leverages the motion priors available in STORM to establish a robust foundation for both CVAC and DCN. To adapt the framework to other feed-forward models, we fine-tune the entire framework for a short budget of 20K steps. During this adaptation, we remove the explicit offset supervision $\mathcal{L}_{\mathrm{dcn}}$ as other models may not provide the necessary motion information, so DCN is instead optimized implicitly through the image reconstruction objectives on $\hat{\mathbf{I}}_{\mathrm{tgt}}^{\mathrm{ref}}$ and $\hat{\mathbf{I}}_{\mathrm{tgt}}^{\mathrm{final}}$. This strategy allows our framework to be efficiently adapted to any feed-forward model with a lightweight fine-tuning schedule.

Table 2: Comparison with diffusion-based refiners on Waymo and nuScenes. Our method achieves the best results in most settings. 1st and 2nd are highlighted.

Method	Waymo			nuScenes		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
STORM + Difix3D+	26.21	0.793	0.078	22.99	0.661	0.137
STORM + ArtiFixer	25.48	0.774	0.099	23.35	0.666	0.156
STORM + GSFixer	26.79	0.815	0.125	23.00	0.654	0.194
STORM + Ours	**28.32**	**0.850**	**0.074**	**25.37**	**0.740**	**0.134**
DGGT + Difix3D+	26.11	0.788	0.073	**25.85**	0.794	**0.084**
DGGT + ArtiFixer	26.95	0.819	0.152	21.85	0.781	0.122
DGGT + GSFixer	26.68	0.813	0.113	21.83	0.814	0.119
DGGT + Ours	**27.10**	**0.825**	**0.072**	24.32	**0.818**	0.095
DrivingForward + Difix3D+	25.32	0.746	0.081	26.22	0.780	0.076
DrivingForward + ArtiFixer	24.85	0.780	0.116	27.34	0.792	0.065
DrivingForward + GSFixer	25.33	0.762	0.092	27.66	0.850	0.065
DrivingForward + Ours	**28.07**	**0.809**	**0.080**	**29.83**	**0.852**	**0.062**

4 Experiment

4.1 Experimental Setup

Datasets and Metrics. We evaluate our method on Waymo Open Dataset (Sun et al., 2020) and nuScenes (Caesar et al., 2020) under the standard driving novel-view synthesis setting. Waymo contains 1,000 driving sequences, each approximately 20 seconds long at 10 FPS, with 798 for training and 202 for validation. nuScenes contains 1,000 urban driving scenes, each about 20 seconds long, split into 700/150/150 for training, validation, and testing. We report results on both validation sets using PSNR, SSIM, and LPIPS.

Baselines and protocol. We evaluate our refinement framework on top of three representative feed-forward driving reconstruction models, namely STORM (Yang et al., 2025), DGGT (Chen et al., 2025a), and DrivingForward (Tian et al., 2025). For all models, we follow their original evaluation protocols for fair comparison. Specifically, STORM and DGGT take frames $\{0, 5, 10, 15\}$ as input and predict frames 0–19, and DrivingForward follows an interpolation protocol that uses frames $t-1$ and $t+1$ as input to predict frame t over the full sequence. We compare against two types of baselines: (i) the original output of each feed-forward model, and (ii) three diffusion-based refiners, Difix3D+ (Wu et al., 2025), ArtiFixer (de Lutio et al., 2026), and GSFixer (Yin et al., 2025). For a fair comparison, we fine-tune the diffusion-based refiners with extended training schedules to ensure adequate convergence, as detailed in the appendix.

Implementation details. We train our framework and all three diffusion-based refiners solely on Waymo, and evaluate them on the validation sets of Waymo and nuScenes to assess cross-dataset generalization. For our method, we first pre-train on STORM (Yang et al., 2025) for 100K steps using the full objective in Eq. (8), taking approximately one day. We then adapt the framework to DGGT and DrivingForward by fine-tuning for 20K steps on each model. We use the AdamW optimizer with a learning rate of 1×10^{-4} and a batch size of 2 on 8 NVIDIA H20 GPUs. The loss weights are set to $\lambda_{\text{ref}} : \lambda_{\text{final}} : \lambda_{\text{dcn}} = 1 : 0.3 : 0.2$. For a fair comparison, all three diffusion-based refiners are fine-tuned for 40K steps on the same training data.

4.2 Main Results

Generalization across Models. We compare each feed-forward model with and without our refinement framework on Waymo and nuScenes, as reported in Tab. 1. Across all three models, our method consistently improves PSNR, SSIM, and LPIPS on both datasets, with qualitative gains visualized in Fig. 4. Notably, STORM reaches 28.32 dB PSNR on Waymo and DrivingForward reaches

Table 3: Component ablation on STORM, including evaluation on dynamic regions.

Method	PSNR ↑	SSIM ↑	LPIPS ↓
Base STORM	25.81	0.804	0.099
w/o CVRC	27.01	0.824	0.124
w/o CVAC	27.27	0.841	0.078
Ours	28.32	0.850	0.074
Dynamic Region			
w/o dcn	23.50	0.686	–
Ours	23.80	0.697	–

Table 4: Efficiency comparison across refiners.

Metric	Ours	Difix3D+	ArtiFixer	GSFixer
Params ↓	**3 M**	1.3 B	1.3 B	5 B
FPS ↑	**162.70**	17.80	3.95	1.36

Table 5: Per-branch gains (Δ) across opacity regions.

	Low-$\hat{A}$		High-$\hat{A}$	
Method	PSNR ↑	SSIM ↑	PSNR ↑	SSIM ↑
CVAC only	2.27	0.010	0.90	0.011
CVRC only	0.94	0.009	2.18	0.019
Full	3.01	0.018	2.62	0.025

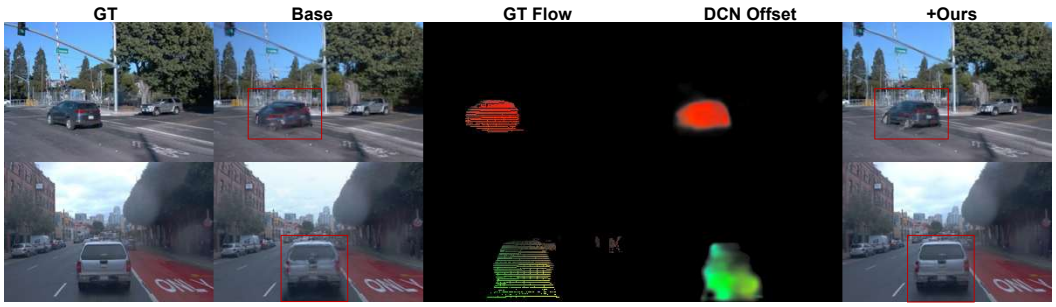

Figure 3: Visualization of DCN offsets. Waymo official GT flow is shown for visual reference. DCN successfully focuses its corrections on moving objects, improving the rendering after refinement.

29.83 dB PSNR on nuScenes, both setting a new state-of-the-art on the respective benchmarks. Despite their substantial differences in architecture and inductive biases, all three models benefit from the framework to a comparable degree, demonstrating that it generalizes broadly across feed-forward driving reconstruction models. While trained exclusively on Waymo, this solution yields consistent improvements on nuScenes, indicating its strong generalization across diverse driving environments.

Comparison with Diffusion-based Refiners. We compare our framework against Difix3D+ (Wu et al., 2025), ArtiFixer (de Lutio et al., 2026), and GSFixer (Yin et al., 2025). As shown in Tab. 2, our approach outperforms all three refiners on Waymo across all metrics. While Difix3D+ remains competitive on DGGT under nuScenes, our method maintains a clear lead on STORM and DrivingForward. Qualitative results are illustrated in Fig. 4. Beyond rendering quality, our framework is substantially more efficient, requiring only 3M parameters and achieving over $9\times$ higher FPS than even single-step diffusion refinement, as detailed in Tab. 4. Overall, these results demonstrate that our method achieves strong generalization across diverse models and datasets with substantially lower computational overhead.

4.3 Ablation Study

Component Ablation. We perform an ablation study of our two main components on STORM as detailed in Tab. 3 (Refer to Appendix for additional ablations). Removing CVRC leads to a performance drop across all three metrics compared to the full model. Specifically, the higher LPIPS than the base model arises from CVAC focusing on low-frequency appearance correction, which may slightly smooth textures that are ultimately restored by CVRC's high-frequency structural correction. Similarly, discarding CVAC also results in degradation across all evaluation metrics. The full model achieves the best performance, proving that these two modules are complementary and their combination is essential for the best performance.

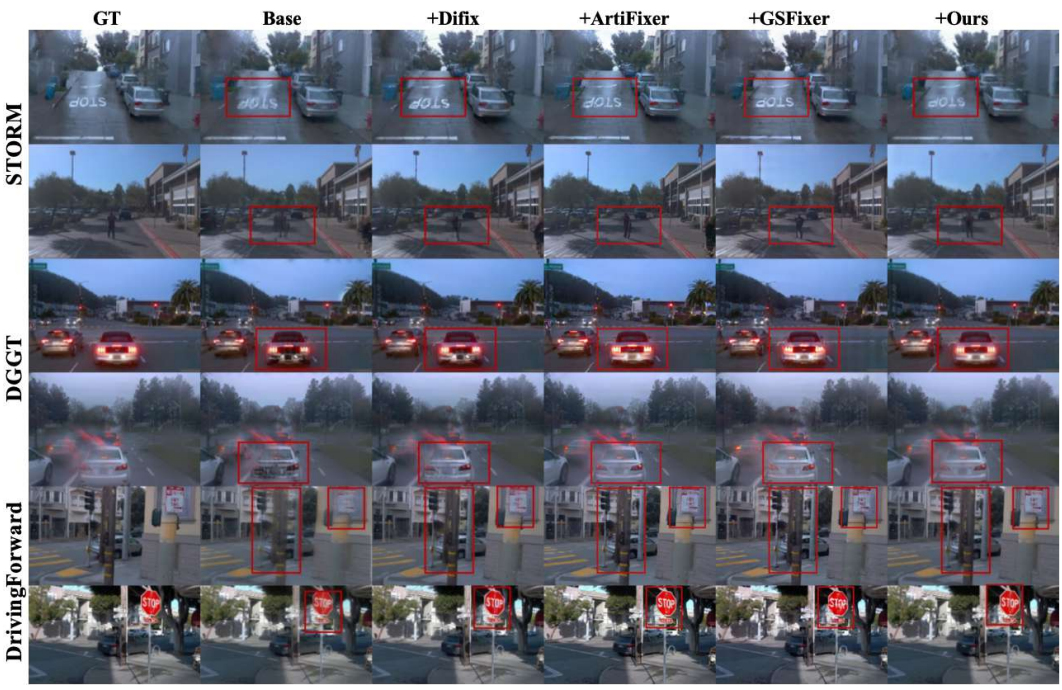

Figure 4: Qualitative comparison on Waymo. Our refinement improves the base rendering and achieves better visual quality than diffusion-based refiners. (See Appendix for more visualizations.)

Dynamic Correction. We evaluate the effectiveness of DCN by removing it from the framework and testing on dynamic regions, as reported in the bottom block of Tab. 3. Its removal leads to a consistent drop in both PSNR and SSIM, confirming its contribution to refining moving objects. We omit LPIPS here as it is not a pixel-level metric. Fig. 3 further visualizes the learned offsets, showing that the sampling corrections are meaningfully concentrated on moving objects.

Opacity-aware Routing. To verify that our opacity-aware routing effectively specializes across different error modes, we analyze the per-branch gains over the STORM baseline across low-$\hat{A}$ and high-$\hat{A}$ regions, as reported in Tab. 5. In low-$\hat{A}$ regions, CVAC yields a $+2.27$ dB PSNR gain, considerably outperforming the $+0.94$ dB gain from CVRC. This pattern reverses in high-$\hat{A}$ regions, where CVRC provides a $+2.18$ dB gain while CVAC contributes only $+0.90$ dB. These complementary results validate our design of routing signals by accumulated opacity and confirm that each branch specializes in its intended error mode.

5 Conclusion

We present AlphaRefine, a lightweight and generalizable post-hoc refinement framework for feed-forward driving scene reconstruction. Motivated by opacity-correlated rendering errors, we design an opacity-routed dual-branch framework that exploits in-context source cues through CVAC and CVRC. This approach adaptively applies targeted corrections without altering the underlying backbone. Extensive experiments on Waymo and nuScenes demonstrate that our method consistently improves multiple feed-forward models, establishing new state-of-the-art performance on both benchmarks. Despite these achievements, several practical limitations remain. The refinement quality is inherently tied to the backbone's representation capacity, and CVRC's residual transport is constrained by source-view visibility in occluded or out-of-view regions. We plan to address these challenges through multi-source-view transport or lightweight generative inpainting in future work.

AI USE STATEMENT

In this work,a large language model was used to assist with manuscript organization, language editing, and LaTeX formatting. The authors reviewed and verified all AI-assisted content and take responsibility for the final manuscript, including all scientific claims, equations, citations, experimental results, and conclusions.

ETHICS STATEMENT

This work aims to improve the visual quality of driving scene reconstruction using existing models and publicly available datasets, in accordance with their respective licenses and usage terms. As driving scene imagery may contain privacy-sensitive information, appropriate care should be taken when handling and sharing such data. While the proposed method may benefit autonomous driving simulation, its outputs should be validated before use in safety-critical applications.

REPRODUCIBILITY STATEMENT

We provide the information needed to reproduce our method and experiments. The model architecture is detailed in Appendix A, while the training objectives and adaptation strategy are described in Sec. 3.4. The Implementation Details paragraph specifies the datasets, evaluation protocols, baselines, and training settings. Additional implementation details, including network configurations, loss functions, source-view selection, and optimization settings, are provided in Appendix B.

REFERENCES

Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuScenes: A multimodal dataset for autonomous driving. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Xiaoxue Chen, Ziyi Xiong, Yuantao Chen, Gen Li, Nan Wang, Hongcheng Luo, Long Chen, Haiyang Sun, Bing Wang, Guang Chen, et al. Dggt: Feedforward 4d reconstruction of dynamic driving scenes using unposed images. *arXiv preprint arXiv:2512.03004*, 2025a.

Yurui Chen, Chun Gu, Junzhe Jiang, Xiatian Zhu, and Li Zhang. Periodic vibration Gaussian: Dynamic urban scene reconstruction and real-time rendering. *arXiv preprint arXiv:2311.18561*, 2023.

Ziyu Chen, Jiawei Yang, Jiahui Huang, Riccardo de Lutio, Janick Martinez Esturo, Boris Ivanovic, Or Litany, Zan Gojcic, Sanja Fidler, Marco Pavone, Li Song, and Yue Wang. OmniRe: Omni urban scene reconstruction. In *International Conference on Learning Representations (ICLR)*, 2025b.

Riccardo de Lutio, Tobias Fischer, Yen-Yu Chang, Yuxuan Zhang, Jay Zhangjie Wu, Xuanchi Ren, Tianchang Shen, Katarina Tothova, Zan Gojcic, and Haithem Turki. Artifixer: Enhancing and extending 3d reconstruction with auto-regressive diffusion models. In *SIGGRAPH*, 2026.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics (SIGGRAPH)*, 42(4), 2023.

Hao Lu, Tianshuo Xu, Wenzhao Zheng, Yunpeng Zhang, Wei Zhan, Dalong Du, Masayoshi Tomizuka, Kurt Keutzer, and Ying-Cong Chen. Drivingrecon: Large 4d gaussian reconstruction model for autonomous driving. In *Advances in Neural Information Processing Systems*, 2025.

Sheng Miao, Jiaxin Huang, Dongfeng Bai, Xu Yan, Hongyu Zhou, Yue Wang, Bingbing Liu, Andreas Geiger, and Yiyi Liao. EVolSplat: Efficient volume-based Gaussian splatting for urban view synthesis. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision (ECCV)*, 2020.

Julian Ost, Fahim Mannan, Nils Thuerey, Julian Knodt, and Felix Heide. Neural scene graphs for dynamic scenes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

Konstantinos Rematas, Andrew Liu, Pratul P. Srinivasan, Jonathan T. Barron, Andrea Tagliasacchi, Thomas Funkhouser, and Vittorio Ferrari. Urban radiance fields. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Xuejian Rong, Jia-Bin Huang, Ayush Saraf, Changil Kim, and Johannes Kopf. Boosting view synthesis with residual transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Chen Shi, Shaoshuai Shi, Xiaoyang Lyu, Chunyang Liu, Kehua Sheng, Bo Zhang, and Li Jiang. Unisplat: Unified spatio-temporal fusion via 3d latent scaffolds for dynamic driving scene reconstruction. In *International Conference on Learning Representations*, 2026.

Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Matthew Tancik, Vincent Casser, Xinchen Yan, Sabeek Pradhan, Ben Mildenhall, Pratul P. Srinivasan, Jonathan T. Barron, and Henrik Kretzschmar. Block-NeRF: Scalable large scene neural view synthesis. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Qijian Tian, Xin Tan, Yuan Xie, and Lizhuang Ma. Drivingforward: Feed-forward 3d gaussian splatting for driving scene reconstruction from flexible surround-view input. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 7374–7382, 2025. doi: 10.1609/aaai.v39i7.32793.

Jingkang Wang, Henry Che, Yun Chen, Ze Yang, Lily Goli, Sivabalan Manivasagam, and Raquel Urtasun. Flux4D: Flow-based unsupervised 4D reconstruction. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Jay Zhangjie Wu, Yuxuan Zhang, Haithem Turki, Xuanchi Ren, Jun Gao, Mike Zheng Shou, Sanja Fidler, Zan Gojcic, and Huan Ling. Difix3d+: Improving 3d reconstructions with single-step diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 26024–26035, 2025.

Ziyang Xie, Junge Zhang, Wenye Li, Feihu Zhang, and Li Zhang. S-NeRF: Neural radiance fields for street views. In *International Conference on Learning Representations (ICLR)*, 2023.

Yunzhi Yan, Haotong Lin, Chenxu Zhou, Weijie Wang, Haiyang Sun, Kun Zhan, Xianpeng Lang, Xiaowei Zhou, and Sida Peng. Street gaussians: Modeling dynamic urban scenes with gaussian splatting. In *European Conference on Computer Vision*, 2024.

Jiawei Yang, Boris Ivanovic, Or Litany, Xinshuo Weng, Seung Wook Kim, Boyi Li, Tong Che, Danfei Xu, Sanja Fidler, Marco Pavone, and Yue Wang. EmerNeRF: Emergent spatial-temporal scene decomposition via self-supervision. In *International Conference on Learning Representations (ICLR)*, 2024.

Jiawei Yang, Jiahui Huang, Boris Ivanovic, Yuxiao Chen, Yan Wang, Boyi Li, Yurong You, Apoorva Sharma, Maximilian Igl, Peter Karkus, Danfei Xu, Yue Wang, and Marco Pavone. Storm: Spatio-temporal reconstruction model for large-scale outdoor scenes. In *International Conference on Learning Representations*, 2025.

Ze Yang, Yun Chen, Jingkang Wang, Sivabalan Manivasagam, Wei-Chiu Ma, Anqi Joyce Yang, and Raquel Urtasun. UniSim: A neural closed-loop sensor simulator. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Xingyilang Yin, Qi Zhang, Jiahao Chang, Ying Feng, Qingnan Fan, Xi Yang, Chi-Man Pun, Huaqi Zhang, and Xiaodong Cun. Gsfixer: Improving 3d gaussian splatting with reference-guided video diffusion priors. *arXiv preprint arXiv:2508.09667*, 2025.

Xiaoyu Zhou, Zhiwei Lin, Xiaojun Shan, Yongtao Wang, Deqing Sun, and Ming-Hsuan Yang. Drivinggaussian: Composite gaussian splatting for surrounding dynamic autonomous driving scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

APPENDIX

A DETAILED NETWORK ARCHITECTURE

This section provides additional architectural details for the two trainable components in our framework: Cross-View Appearance Correction (CVAC) and the Dynamic Correction Network (DCN).

A.1 CVAC: CROSS-VIEW APPEARANCE CORRECTION

CVAC is an image-space refinement network that leverages source-view appearance cues to correct low-frequency rendering artifacts. It consists of a shared RGB encoder, an opacity encoder, a cross-attention block for target-to-source feature aggregation, and a residual decoder head.

Shared RGB Encoder (E_{rgb}). Both the initial target rendering $\hat{\mathbf{I}}_{\mathrm{tgt}}$ and the source-view image $\mathbf{I}_{\mathrm{src}}$ are processed by a shared encoder, so that they are mapped into the same feature space. The encoder consists of an initial 3×3 convolution followed by two residual blocks, maintaining a constant feature dimension of 64. Each residual block uses GroupNorm with 32 groups and Swish activation.

Opacity Encoder (E_{α}). To provide the network with the base model's rendering confidence, the accumulated opacity $\hat{\mathbf{A}}_{\mathrm{tgt}}$ is encoded by a separate lightweight branch. This encoder applies a 3×3 convolution with 16 output channels, a GELU activation, and a 1×1 convolution that projects the opacity feature to 64 channels.

Cross-Attention Block. The core of CVAC is a target-to-source cross-attention mechanism. For computational efficiency, the feature maps are downsampled by a factor of 4 using average pooling before the attention operation. We use 4 attention heads with a per-head dimension of 16. The query, key, and value projections are implemented as 1×1 convolutions, with LayerNorm applied to the queries and keys for stable training. The attention output is then bilinearly upsampled to the original resolution and fused with the target feature.

Residual Decoder Head. The target feature, cross-attended source feature, and opacity feature are concatenated, resulting in a 192-channel feature map. A 1×1 convolution first reduces the channel dimension to 64, followed by a residual block and a final output convolution that predicts the RGB correction $\Delta\mathbf{I}$. The weights and biases of the final convolution are initialized to zero, so that CVAC starts from an identity mapping and preserves the base rendering at the beginning of training. The layer configurations are summarized in Tab. 6.

Table 6: Architectural details of the CVAC module.

Component	Layer / Operation	Kernel	Output Channels
Shared RGB Encoder	Conv2d + 2 × ResBlock	3×3	64
Opacity Encoder	Conv2d + GELU + Conv2d	3×3, 1×1	64
Cross-Attention	4-head Multi-head Attention	1×1	64
Fusion Layer	Concat + Conv2d + ResBlock	1×1	64
Residual Head	GroupNorm + Swish + Conv2d	3×3	3

A.2 DCN: DYNAMIC CORRECTION NETWORK

DCN is a lightweight U-Net designed to predict pixel-level sampling offsets for CVRC residual transport. It corrects geometric mismatches between source and target views, especially in dynamic regions where static reprojection is unreliable.

Input and Output Formulation. DCN takes a 9-channel input formed by concatenating the source image, the target rendering reprojected into the source view, and their absolute difference. The network outputs a 2-channel offset field $\delta\mathbf{p} = (\Delta u, \Delta v)$. The output is constrained by a $\tanh$

Table 7: Architectural details of the DCN U-Net.

Stage	Operations	Channels	Resolution
Input	Image concatenation	9	$H \times W$
Encoder 1	ConvBlock + Stride-2 Conv	32	$H/2 \times W/2$
Encoder 2	ConvBlock + Stride-2 Conv	64	$H/4 \times W/4$
Encoder 3	ConvBlock + Stride-2 Conv	128	$H/8 \times W/8$
Bottleneck	ConvBlock	256	$H/8 \times W/8$
Decoder 1	Upsample + Concat + ConvBlock	128	$H/4 \times W/4$
Decoder 2	Upsample + Concat + ConvBlock	64	$H/2 \times W/2$
Decoder 3	Upsample + Concat + ConvBlock	32	$H \times W$
Output Head	Conv2d + $\tanh \times d_{\max}$	2	$H \times W$

activation and scaled by a maximum displacement hyperparameter $d_{\max}$, which bounds the sampling correction range.

U-Net Architecture. The network follows a 3-stage encoder-decoder architecture with skip connections to preserve spatial details. The hidden dimensions across the encoder stages are 32, 64, 128, and 256. Each stage uses a ConvBlock consisting of two 3×3 convolutions with GroupNorm and SiLU activation. Downsampling is performed using stride-2 convolutions, while upsampling uses bilinear interpolation followed by feature concatenation and convolutional refinement. The architecture is summarized in Tab. 7.

B Training and Implementation Details

We provide additional implementation details for our refinement framework. For all experiments, the underlying feed-forward reconstruction model is kept frozen, and only the refinement modules, i.e., CVAC and DCN, are optimized. CVRC itself is a geometry-based residual transport procedure and contains no learnable parameters. All refinement modules operate at a resolution of 160×240, following the official STORM setting. For base models that render at a different resolution, we first run their original rendering pipeline and then resize the rendered RGB image, depth, accumulated opacity, source images, and ground-truth images to 160×240 before refinement and metric computation.

Training objective. The refinement framework is trained with an intermediate reconstruction loss on the CVAC output, a final reconstruction loss on the full refined output, and an offset supervision loss for DCN when available:

$$\mathcal{L} = 1.0\,\mathcal{L}_{\text{ref}} + 0.3\,\mathcal{L}_{\text{final}} + 0.2\,\mathcal{L}_{\text{dcn}}. \tag{9}$$

Both image reconstruction terms use the same combination of pixel, structural, and perceptual losses:

$$\mathcal{L}_{\text{rec}} = 0.8\,\mathcal{L}_1 + 0.2\,\mathcal{L}_{\text{SSIM}} + 0.2\,\mathcal{L}_{\text{perc}}. \tag{10}$$

Here, $\mathcal{L}_{\text{ref}} = \mathcal{L}_{\text{rec}}(\hat{\mathbf{I}}^{\text{ref}}, \mathbf{I})$ and $\mathcal{L}_{\text{final}} = \mathcal{L}_{\text{rec}}(\hat{\mathbf{I}}^{\text{out}}, \mathbf{I})$.

On STORM, we additionally supervise DCN with pseudo offsets derived from STORM's predicted 3D motion. For each target pixel, we compute two target-to-source correspondences: a static correspondence $\mathbf{p}_s^{\text{static}}$ obtained by projecting the depth-lifted target point to the selected source view, and a motion-aware correspondence $\mathbf{p}_s^{\text{motion}}$ obtained by first moving the lifted 3D point using STORM's predicted 3D motion and then projecting it to the same source view. The pseudo offset is defined as

$$\delta\mathbf{p}^{\text{pseudo}} = \mathbf{p}_s^{\text{motion}} - \mathbf{p}_s^{\text{static}}. \tag{11}$$

The DCN offset loss is then

$$\mathcal{L}_{\text{dcn}} = \text{SmoothL1}\left(\delta\mathbf{p}, \delta\mathbf{p}^{\text{pseudo}}\right), \tag{12}$$

where $\delta\mathbf{p}$ denotes the offset predicted by DCN. Invalid projections, including out-of-bound samples or points projected behind the source camera, are excluded from this loss. When adapting to DGGT and DrivingForward, which do not provide compatible 3D motion supervision, we remove $\mathcal{L}_{\text{dcn}}$ and optimize DCN implicitly through the final image reconstruction loss.

Source-view selection. CVAC uses all available source/context frames as appearance references, but only aggregates features within the same camera stream to avoid mixing camera-dependent appearance statistics. Specifically, source features from the same camera are temporally averaged and used as keys and values in the cross-attention block, while the target rendering feature serves as the query. CVRC uses a single source frame for residual transport. Unless otherwise specified, we select the temporally nearest context frame and construct the source residual as the difference between the source-view ground truth and the corresponding refined source rendering. Invalid reprojections, including out-of-bound or behind-camera samples, are masked out and receive zero residual correction.

STORM. For STORM, we use the official checkpoint and the official 3-camera NVS protocol. The context frames are sampled as $\{s, s+5, s+10, s+15\}$ from a 2-second window, which typically contains 20 frames in Waymo. The refinement loss is computed on non-context target frames. STORM is kept frozen and provides the initial rendering, depth, accumulated opacity, and 3D motion used for DCN pseudo-offset supervision. All STORM-based refinement experiments use the official 160×240 resolution.

DGGT and DrivingForward. For DGGT, we use the official released 1-view checkpoint with the front camera and evaluate under its official NVS setting. DGGT is frozen and only provides the base rendering, depth, opacity, and geometry required by the refinement modules. For DrivingForward, we use the official checkpoint with the front 3 cameras. Its NVS setting uses adjacent frames $t-1$ and $t+1$ as context to render the center target frame t. DrivingForward is also kept frozen. Since DrivingForward renders at a higher native resolution, we resize its outputs and ground truth to 160×240 after rendering, before applying refinement and computing metrics.

Optimization and comparison with diffusion-based refiners. We train our refinement modules using AdamW with a learning rate of 1×10^{-4}, weight decay 1×10^{-4}, a constant learning-rate schedule, and global gradient clipping with a maximum norm of 1.0. The batch size is 2 per GPU. For the diffusion-based baselines, we fine-tune the official pretrained checkpoints of Difix3D+ (Wu et al., 2025) and the 1.3B variant of ArtiFixer (de Lutio et al., 2026). For GSFixer (Yin et al., 2025), we fine-tune its CogVideoX-5B-I2V-based restoration model. We follow each method's official training procedure, retaining its original conditioning strategy and optimization objectives. All three refiners are fine-tuned for 40K steps on the same Waymo training data used by our method, with matched source and target views and base-model renderings. We evaluate all methods on the same validation frames at the same image resolution to ensure a consistent comparison protocol.

C Details of Opacity-dependent Error Analysis

We provide additional details on the opacity-dependent frequency error analysis used in main paper. The same analysis protocol is applied to all evaluated feed-forward base models. For each model, we use its frozen renderer to obtain the initial rendering $\hat{\mathbf{I}}$, the ground-truth image $\mathbf{I}$, and the accumulated opacity map $\hat{\mathbf{A}} \in [0, 1]$. The analysis does not use dynamic masks, depth, or optical flow; it only relies on the ground-truth image, the rendered image, and the accumulated opacity map.

Opacity-stratified regions. For each rendered image, we partition pixels according to the percentile of the accumulated opacity $\hat{\mathbf{A}}$. Specifically, we define the low-opacity region as the bottom 30% pixels and the high-opacity region as the top 30% pixels. The middle 40% pixels are not used in the main comparison. We use percentile-based partitions rather than a fixed opacity threshold to avoid bias caused by different opacity calibration across base models. This opacity partition is used only for analysis; our method itself uses the continuous opacity value as a soft routing signal.

Frequency decomposition. Let the RGB rendering residual be

$$\mathbf{E} = \mathbf{I} - \hat{\mathbf{I}}. \tag{13}$$

We decompose the residual into low-frequency and high-frequency components using an average-pooling low-pass filter:

$$\mathbf{E}_{\text{low}} = \text{AvgPool}_{k=9}(\mathbf{E}), \qquad \mathbf{E}_{\text{high}} = \mathbf{E} - \mathbf{E}_{\text{low}}. \tag{14}$$

Table 8: Additional ablation studies on DGGT and DrivingForward. The upper block reports component ablations, and the lower block reports the effect of DCN on dynamic regions.

	DGGT			DrivingForward		
Method	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Base	24.64	0.799	0.104	24.85	0.764	0.116
CVRC only	25.15	0.796	0.093	25.88	0.776	0.095
CVAC only	26.59	0.820	0.078	27.18	0.789	0.102
Full	27.10	0.825	0.072	28.07	0.809	0.080
Dynamic Region						
Full w/o DCN	23.42	0.668	–	22.64	0.682	–
Full	23.67	0.680	–	23.15	0.719	–

We use reflection padding for the low-pass filtering to avoid artificial boundary artifacts. For each opacity region, we compute the mean absolute low-frequency error, mean absolute high-frequency error, and the corresponding frequency ratios:

$$r_{\text{low}} = \frac{\text{mean}(|\mathbf{E}_{\text{low}}|)}{\text{mean}(|\mathbf{E}|)}, \qquad r_{\text{high}} = \frac{\text{mean}(|\mathbf{E}_{\text{high}}|)}{\text{mean}(|\mathbf{E}|)}. \tag{15}$$

We additionally report the high-frequency MSE ratio,

$$r_{\text{high}}^{\text{mse}} = \frac{\text{mean}(\mathbf{E}_{\text{high}}^2)}{\text{mean}(\mathbf{E}^2)}, \tag{16}$$

which emphasizes high-frequency structural discrepancies.

Luminance, chroma, and structural errors. To separately measure low-frequency brightness and color deviations, we convert RGB residuals to the BT.601 YCbCr color space:

$$Y = 0.299R + 0.587G + 0.114B, \tag{17}$$

$$C_b = -0.168736R - 0.331264G + 0.5B, \quad C_r = 0.5R - 0.418688G - 0.081312B. \tag{18}$$

The low-frequency luminance error is computed as $\text{mean}(|Y_{\text{low}}|)$, while the low-frequency chroma error is computed as

$$\text{mean}\left(\sqrt{C_{b,\text{low}}^2 + C_{r,\text{low}}^2}\right). \tag{19}$$

To measure structural errors, we compute a Sobel gradient residual on the luminance channel:

$$\mathbf{E}_{\text{sobel}} = \sqrt{\left(\nabla_x Y - \nabla_x \hat{Y}\right)^2 + \left(\nabla_y Y - \nabla_y \hat{Y}\right)^2}. \tag{20}$$

This metric captures edge and boundary discrepancies, which are representative high-frequency structural errors.

Reported statistics. For the motivation analysis, we mainly compare the bottom-30% and top-30% opacity regions. We report the low-frequency error ratio in low-opacity regions, the high-frequency error ratio in high-opacity regions, the relative low-frequency luminance and chroma errors between low- and high-opacity regions, and the relative high-frequency MSE and Sobel gradient errors between high- and low-opacity regions. All statistics are averaged over the evaluated validation samples for each base model. The results consistently show that low-opacity regions are dominated by low-frequency appearance shifts, whereas high-opacity regions contain stronger high-frequency and structural errors.

D Additional Ablation Studies

Component analysis on additional models. In the main paper, we report component ablations on STORM. To further examine whether the two refinement branches remain effective across different feed-forward reconstruction models, we additionally conduct component analysis on DGGT

Table 9: Region-wise branch behavior on DGGT. CVAC provides larger gains in low-$\hat{A}$ regions, while CVRC is more effective in high-$\hat{A}$ regions.

	Low-$\hat{A}$ Region				High-$\hat{A}$ Region			
Method	PSNR ↑	SSIM ↑	ΔPSNR ↑	ΔSSIM ↑	PSNR ↑	SSIM ↑	ΔPSNR ↑	ΔSSIM ↑
Base	16.95	0.705	0.00	0.000	29.34	0.833	0.00	0.000
CVAC only	19.84	0.757	2.89	0.052	29.36	0.837	0.02	0.004
CVRC only	17.30	0.706	0.35	0.001	29.84	0.839	0.50	0.006
Full	20.05	0.758	3.10	0.053	29.85	0.841	0.51	0.008

Table 10: Region-wise branch behavior on DrivingForward. CVAC mainly improves low-$\hat{A}$ regions, while CVRC brings larger gains in high-$\hat{A}$ regions.

	Low-$\hat{A}$ Region				High-$\hat{A}$ Region			
Method	PSNR ↑	SSIM ↑	ΔPSNR ↑	ΔSSIM ↑	PSNR ↑	SSIM ↑	ΔPSNR ↑	ΔSSIM ↑
Base	22.43	0.766	0.00	0.000	27.74	0.749	0.00	0.000
CVAC only	25.50	0.799	3.07	0.033	28.79	0.769	1.05	0.020
CVRC only	23.28	0.776	0.85	0.010	29.19	0.781	1.45	0.032
Full	25.76	0.805	3.33	0.039	29.62	0.788	1.88	0.039

and DrivingForward. As shown in Tab. 8, the relative contribution of each branch is backbone-dependent. CVAC provides stronger global PSNR gains on these two models, suggesting that their dominant errors are more related to low-frequency appearance shifts. Meanwhile, CVRC still provides complementary improvements, especially in perceptual quality, and the full framework achieves the best overall performance on both backbones.

Region-wise branch behavior. We further analyze the branch behavior in low- and high-opacity regions. As shown in Tab. 9 and Tab. 10, the errors of DGGT and DrivingForward are largely concentrated in low-$\hat{A}$ regions, where the base renderings have much lower PSNR than in high-$\hat{A}$ regions. As a result, CVAC, which targets low-opacity appearance correction, yields larger overall gains than CVRC on these two backbones. This does not contradict our opacity-routed design: within each opacity regime, the branch behavior remains consistent with our motivation. CVAC provides the dominant improvement in low-$\hat{A}$ regions, while CVRC brings larger gains than CVAC in high-$\hat{A}$ regions. The full model achieves the best performance in both regimes, further validating the complementary design of CVAC and CVRC.

Effect of DCN. We further evaluate the effect of DCN, which predicts dynamic-aware sampling offsets for CVRC. Without DCN, residual transport relies only on static geometric correspondences, which can be inaccurate when the target and source views contain moving objects. As shown in Tab. 8, adding DCN improves PSNR and SSIM on both DGGT and DrivingForward, indicating that the learned offset provides useful correction for dynamic-aware residual transport.

E Additional Qualitative Results

We provide additional qualitative comparisons between the base renderings and our refined results on different feed-forward backbones. Each figure shows the ground truth, the base rendering, and our refined result. These examples further demonstrate that our method improves appearance consistency and local structural details across different driving reconstruction models.



Figure 5: Additional qualitative results on STORM. Compared with the base rendering, our method improves low-frequency appearance consistency and recovers sharper local details.



Figure 6: Additional qualitative results on DGGT.



Figure 7: Additional qualitative results on DrivingForward.